Baptiste Provendier, bpp2218, 01553706

1. CNN Introduction: task 1

The first parameter that was tested was the number of convolutional layers in our network. The following settings were used for each layer: Conv2D(32, (3,3), padding='same', strides=2) with a ReLu activation function.The results show that using 5 layers is optimal as there is no discrepancy between the training and testing error, allowing space to add complexity to the model.

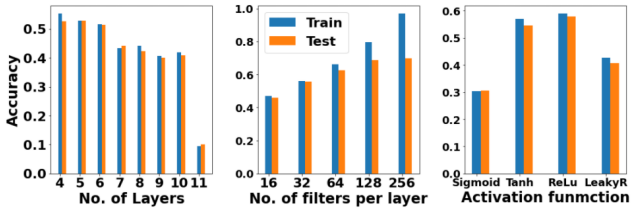

Figure 1: Charts illustrating the effects of the most influential parameters on the model accuracy

In Figure 1, it is clear that increasing the number of filters per layer contributes to greater training accuracy. The greater the number of filters, the more features are extracted per image which increases the complexity of the model. However, a more complex model can lead to over fitting which negatively affect the test accuracy and it is shown by the large discrepancy between the training and testing data. Using more than 10 layers highlights the issue of using the softmax activation function and cross entropy loss in an unregularised model [1]. Once the network outputs predictions of 0.0, the loss contains a value of log(0) or NaN. Any operation performed on NaN is also NaN which quickly backpropagates through the network forcing the same prediction for any input [2]. Lastly, the tanh and ReLu activation functions show the best results but ReLu is preferred because it is more computationally efficient [3].

The chosen model will then contain 5 layers with a decreasing number of filters per layer, a (4x4) filter size for each filter with 3 max pooling layers. (128, 64, 64, MaxPool, 32, MaxPool, 32, MaxPool, Dense(64), Dense(32), Dense(10)) as illustrated in Figure2. This model achieved a training accuracy of 0.7898 and a test accuracy of 0.7348.

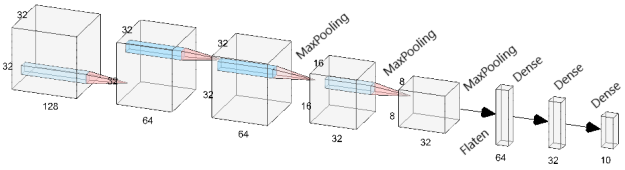

Figure 2: Model Architecture for the chosen model [4]

2. CNN Introduction: task 2

For this task, let's compare the errors for three models illustrated in Figure 3. A simple design (4,4), a complex one (8, 8, 16, 16, 32, 32, 64, 64, 128, 128, 256, 256, 512, 512) and the chosen model (8,Maxpool,16,16,32,Maxpool,64,Maxpool, 128, Dense(32)).

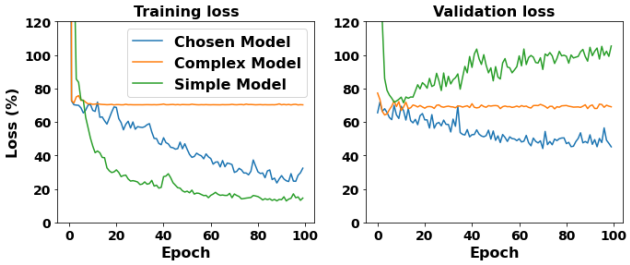

Figure 3: Training and validation losses for 3 architectures

The simple model achieves low training and high validation loss. The model extracts very few features and is trained to recognise the specific images it is shown and generalises very poorly. The complex model shows a flat curve. Higher learning rates will decay loss faster but will converge to a sub-optimal solution in an unstable optimisation landscape [5] [6]. Reducing the learning rate able the chosen model to decrease the training loss with the epochs. The chosen model achieves the following errors based on the different parts of the house.

Location	Estimation error (%)	Training error (%)
Frontal	48.12	24.71
Kitchen	50.66	19.21
Bedroom	46.94	15.93
Bathroom	46.22	25.65

Table 1: Errors per location for the chosen model

References

[1] https://deepnotes.io/softmax-crossentropy

[2] https://stackoverflow.com/questions/44477443/training-and-validation-accuracy-suddenly-drops-in-convolutional-neural-network

[3] http://www.cs.toronto.edu/ fritz/absps/imagenet.pdf

[4] http://alexlenail.me/NN-SVG/AlexNet.html

[5] https://cs231n.github.io/neural-networks-3/loss

[6] https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/